Title

Tri Nguyen

nguyetr9@oregonstate.edu

December 25, 2022

The right thing to do is Dynamic Programming. But right thing is always hard to do. So the less right things to do are

- Approximation in value space $J^*(x)$.

  For this, we have several options.

  - First, to deal with expectation is a stochastic setting, we can just assume a deterministic setting instead, by eliminating all randomness by their mode 1 value for example.

    Then in deterministic setting, for the min operator, we can use brute force, integer programming (discrete), A^* search (discrete), or nonlinear programming (continuous)

- Approximation in policy space

This is weird. In 2.1.4, they learn an approximate $\widetilde{Q}()$ based on approximate $\widetilde{J}$. I thought getting Q from J should be obvious. It may not because if action space is large.

Parameterization can be used in both value approximation and policy approximation. In both case, the setting is very similar to supervised learning setting.

RL methods has many intertwined components.

2.1.6 shows an interesting point. Good an approximation $\widetilde{J}$ is not necessarily closed to J^*. It is good as long as it is *uniformly distant* from J^*. The proposed criterion is that $Q(x,u) - \widetilde{Q}(x,u)$ change gradually as u changes.

Until now, we only talk about deterministic policy.

Good point: in the use of ℓ-lookahead, as ℓ is getting longer, the role of $\widetilde{J}$ diminishes.

A an ℓ-lookahead, we are solving ℓ-stage DP in an exact manner where we assume the terminal cost is approximated by $\widetilde{J}$.

2.3. Now we talk. Problem approximation. This is what is used in MARL currently.

Kindly think about MARL setting where environment is deterministic, policy is deterministic. could be the use of tensor?

So what so-called *policy improvement* is*rollout algorithm*. It only works under either of the following assumptions:

- The base policy is sequentially consistent, i.e., if it generates sequence of states $s_k, s_{k+1}, \ldots, s_N$ starting from x_k , then it also generates a sequence $s_{k+1}, \ldots, s_N$ starting from s_{k+1}.
- The base policy is sequentially improving.

Definition of rollout algorithm: It applies at state x_k the control $\widetilde{u}_k(x_k)$ given by the minimization

$$\widetilde{u}_k(x_k) \triangleq \arg\min_u \widetilde{Q}_k(x_k, u),$$

where $\widetilde{Q}_k(x_k, u)$ is the approximation of the true $Q_k(x_k, u)$ defined by

$$\widetilde{Q}_k(x_k, u) \triangleq g_k(x_k, u) + H_{k+1}(f_k(x_k, u)),$$

where $H_{k+1}(x_{k+1})$ is cost of the base heuristic starting from x_{k+1}.

Okay, let's try to describe all things without using RL terminologies.

1 Setting and Notation

- The system dynamic is governed by $s_{t+1} = f(s_t, a_t, w_t)$, where s_{t+1} denotes state variable at time $t+1$, a_t is the action variable at time t, w_t is random noise.
- a_t is the action chosen by $\mu_t : \mathcal{S} \to \mathcal{A}$
- At each step, there is a reward $g_t(s_t, a_t, \omega_t)$
- The system starts at state s_0.
- Let the system run for N steps.

Denote $J_\pi : \mathcal{S} \to \mathbb{R}$ as a cost function of $\pi = \{\mu_0, \mu_1, \ldots\}$,

$$J_\pi(s_0) \triangleq \mathop{\mathbb{E}}_{\omega_0,\ldots,w_N}\left[\sum_{t=0}^{N} \alpha^{t-1} g_t(s_t, \mu_t(s_t), \omega_t)\right], \quad \text{where} \tag{1}$$
$$s_{t+1} = f(s_t, \mu_t(s_t), \omega_t)$$

The factor $\alpha < 1$ is not important here. It is used in case of infinite horizon, we are having it here so the transition to infinite horizon can be smoother.

Define $J^* : \mathcal{S} \to \mathbb{R}$ as the optimal cost function

$$J^*(s_0) \triangleq \max_\pi J_\pi(s_0) \tag{2}$$

Note that $J^*(s_0)$ as a well-defined function, but it has nothing to do with any particular π. Optimal $J^*(s_0)$ can be obtained by different π at different s_0. However, this is rarely the case. In particular, $\exists \pi^*$ such that

$$J^*(s_0) = J_{\pi^*}(s_0) \quad \forall s_0 \in \mathcal{S}$$

This might because of the so-called *principle of optimality*. Consider deterministic system for simplicity (stochastic system can be treated in the same way), if we run DP, we then arrive at an optimal $J^\star$, and using that $J^\star$, we can find the optimal π^*. That proves the existence of π^*. This is nice as it is a construction proof.

The DP algorithm specifically exploits the principle of optimality to reduce the search space significantly. In a deterministic system, it essentially repeats

$$J_t^*(s_t) = \arg\min_{a_t} \; g_t(s_t, a_t) + J_{t+1}^*(f(s_t, a_t))$$

Hence at each time step, we can define function $\mu_t^* : \mathcal{S} \to \mathcal{A}$, and collection of such functions over t leads to the construction of π^*. It is the same for stochastic function.

Algorithm 1: Dynamic programming 1

Input: System dynamic: $f_t(s, a, \omega), g_t(s, a, \omega)$, and distribution of all ω_t's; starting state s_0 , number of timestep T

Output: $J^\star(s_0)$, and $J_t^\star(s)$ for $0 \le t \le T, s \in \mathcal{S}$

Set $J_T^\star(s) = \mathbb{E}_{\omega_T}\left[g_T(s, \mu_T(s), \omega_T)\right], \quad \forall s \in \mathcal{S}$
for $t = T-1$ *to* 1 **do**
  $J_t^\star(s) = \min_{\mu_t(\cdot)} \mathbb{E}_{\omega_t}\left[g_t(s, \mu_t(s), \omega_t) + \alpha J_{t+1}^\star(f_t(s, \mu_t(s), \omega_t))\right], \quad \forall s \in \mathcal{S}$
end

It's nice that we only need to deal with one ω_t at a time.

Remark 1.1. Hey, we don't run simulation here. Everything is figured out analytically. Also, in many games, ω_t is a deterministic factor.

Remark 1.2. And the DP algorithm can deal with problem satisfies the *principle of optimality* (volume 1). Principle of optimality states that a tail of the optimal sequence of actions is also optimal.

For any function $J : \mathcal{S} \to \mathbb{R}$, define operators

$$(TJ)(s) \triangleq \min_{u \in U(s)} \mathop{\mathbb{E}}_{\omega} [g(s,u,\omega) + \alpha J(f(s,u,\omega))]$$
$$(T_\mu J)(s) \triangleq \mathop{\mathbb{E}}_{\omega} [g(s,\mu(s),\omega) + \alpha J(f(s,\mu(s),\omega))]$$

$T^k J : \mathcal{S} \to \mathbb{R}$ is a function, which can be viewed as the optimal cost function for the k-stage, α discount problem with cost per step g, and terminal cost $\alpha^k J$.

Then, Algorithm 1 can be rewritten as

$$J^{\text{init}}(s) = \mathop{\mathbb{E}}_{\omega_T} [g(s,\omega_T)]$$
$$J^* = T^N J^{\text{init}}$$

Okay, nothing new happens here. It is still just a vanilla DP applying on a finite horizon problem. It will give us an exact solution.

2 Infinite Horizon

Now let us stretch problem to infinite horizon. What would happen? First of all, the so-called policy π becomes a sequence $\pi = \{\mu_0, \mu_1, \ldots\}$. Accordingly, let us define

$$J_\pi(s_0) \triangleq \lim_{N\to\infty} \mathbb{E}\left[\sum_{t=0}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right], \tag{3}$$

$$J^*(s_0) \triangleq \min_\pi J_\pi(s_0) = \min_\pi \lim_{N\to\infty} \mathbb{E}\left[\sum_{t=0}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right], \tag{4}$$

with respect to system dynamics governed by f, g.

As $\alpha < 1$, and the system dynamic is finite, $J_\pi(s_0)$ is finite, so it is a well-defined quantity, and so is $J^*(s_0)$. What would happen with DB in Algorithm 1 now? As it starts from T in finite horizon case, where would it start now? Does it ever converge to anything? And if it does, what would it be?

The short answers are

- We can pick arbitrary T, set an arbitrary initial function for J_T^*.
- Then we run step 3 in Algorithm 1 infinitely many times.

The procedure is guaranteed to converge, and moreover, it produces the optimal value respect to the objective in (3).

Theorem 2.1. *We have following results (Bertsekas 1999, Section 1.1.3)*

1. *(Convergence of the DP Algorithm) Let J^{init} be a zero function, i.e, $J^{init}(s) = 0$, $\forall s \in \mathcal{S}$. Then,*

$$J^* = \lim_{k\to\infty} T^k J^{init},$$

where J^* defined in (4). Moreover, when $\alpha < 1$, J^{init} can be any bounded function.

2. *(Bellman's equation) By definition of operator T, for any function J*

$$(T^k J)(s) = \min_u \mathop{\mathbb{E}}_{\omega} \left[g(s,u,\omega) + \alpha (T^{k-1} J)(f(s,u,\omega))\right],$$

Then taking $k \to \infty$ and use the first result, we obtain

$$J^*(s) = \min_u \mathop{\mathbb{E}}_{\omega} [g(s,u,\omega) + \alpha J^*(f(s,u,\omega))], \tag{5}$$

or shortly, $J^* = TJ^*$.

3. *(Characterization of optimal stationary policies) If policy μ satisfy Bellman equation, then it is optimal.*

The last paragraph in the book comments that: we can dispense stochastic policy in many cases.

Proposition 2.1 (Existence of optimal stationary policy)**.** *A stationary μ is optimal if and only if*

$$TJ^\star = T_\mu J^\star$$

$J^\star$ always exists under "typical setting".

The first result comes from the fact that the tail of summation diminishes. The proof looks nice.

Theorem 2.2 (Convergence of the DP Algorithm)**.** *Let J^{init} be a zero function, i.e, $J^{init}(s) = 0$, $\forall s \in \mathcal{S}$. Then,*

$$J^* = \lim_{k\to\infty} T^k J^{init},$$

where J^* defined in (4). Moreover, when $\alpha < 1$, J^{init} can be any bounded function.

Proof. For some $k > 0$,

$$\begin{aligned}
J_\pi(s_0) &= \lim_{N\to\infty} \mathbb{E}\left[\sum_{t=1}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right] \\
&= \mathbb{E}\left[\sum_{t=1}^{k} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right] + \lim_{N\to\infty}\left(\mathbb{E}\left[\sum_{t=k+1}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right]\right) \\
&= \mathbb{E}\left[\alpha^k J^{\text{init}}(s_k) + \sum_{t=1}^{k-1} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right] + \lim_{N\to\infty}\left(\mathbb{E}\left[\sum_{t=k+1}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right]\right) - \mathbb{E}[\alpha^k J^{\text{init}}(s_k)]
\end{aligned}$$

Take minimizing on both sides w.r.t π,

$$J^*(s_0) = T^k J^{\text{init}}(s_0) + \lim_{N\to\infty}\left(\mathbb{E}\left[\sum_{t=k+1}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right]\right) - \alpha^k\, \mathbb{E}[J^{\text{init}}(s_k)]$$

As $|g(s, a, \omega)| < D$,

$$\left|\lim_{N\to\infty} \mathbb{E}\left[\sum_{t=k+1}^{N} \alpha^{t-1} g(s_t, \mu_t(s_t), \omega_t)\right]\right| \le D\frac{\alpha^k}{1-\alpha}$$

That leads to

$$-D\frac{\alpha^k}{1-\alpha} - \alpha^k\, \mathbb{E}[J^{\text{init}}(s_k)] \le J^*(s_0) - T^k J^{\text{init}}(s_0) \le D\frac{\alpha^k}{1-\alpha} - \alpha^k\, \mathbb{E}[J^{\text{init}}(s_k)]$$

As $\alpha < 1$, taking $k \to \infty$ when $N \to \infty$, we conclude $J^*(s_0) = J^{\text{init}}(s_0)$. ■

Corollary 2.2.1. For a stationary policy $\pi = \{\mu, \mu, \ldots\}$,

$$J_\pi = \lim_{k\to\infty} T_\pi^k J^{\text{init}}$$

is value iteration DP algorithm?

Theorem 2.3. *(Bellman's equation) We have following results (Bertsekas 1999, Section 1.1.3) By definition of operator T, for any function J*

$$(T^k J)(s) = \min_u \mathop{\mathbb{E}}_{\omega} \left[g(s,u,\omega) + \alpha (T^{k-1} J)(f(s,u,\omega)) \right],$$

Then taking $k \to \infty$ and use the first result, we obtain

$$J^*(s) = \min_u \mathop{\mathbb{E}}_{\omega} \left[g(s,u,\omega) + \alpha J^*(f(s,u,\omega)) \right], \tag{6}$$

or shortly, $J^* = TJ^*$. Furthermore, J^* is the unique solution within the class of bounded functions.

Proof. ■

Theorem 2.4 (Characterization of optimal stationary policies)**.** *A stationary policy μ is optimal iff*

$$TJ^* = T_\mu J^*$$

Since $J^* = TJ^*$, and J^* always exists, this implies that a stationary μ always exists. In particular, given J^*, we can define $\mu : \mathcal{S} \to \mathcal{A}$ as

$$\mu(s) \triangleq \arg\min_{u \in \mathcal{A}} \mathbb{E}[g(s,u,\omega) + \alpha J^*(f(s,u,\omega))]$$

By this construction, $T_\mu J^* = TJ^*$, and hence stationary μ is optimal. However, this does **not** imply that all optimal policies are stationary, but rather shrinking our search space to the class of stationary policies.

These 3 theorems 2.2, 2.3, 2.4 are the backbone of all the RL methods.

3 Methods to find the optimal stationary policy

3.1 Value Iteration

As an easy application, Theorem 2.2 leads to,

Corollary 3.0.1 (Value Iteration)**.** Start with arbitrary bounded function J_0, repeat the following

$$J_{k+1} = TJ_k$$

This is essentially the DP algorithm where we obtain an *approximate* J^*, denoted as $\widetilde{J}^*$, by running DP as finitely many times as possible. Having $\widetilde{J}^*$, we can derive an approximate optimal stationary policy $\widetilde{\mu}^*$ using Theorem 2.4, i.e.,

$$\widetilde{\mu}^* \triangleq \arg\min_{u \in \mathcal{A}} \mathbb{E}[g(s,u,\omega) + \alpha \widetilde{J}^*(f(s,u,\omega))]$$

Note that value iteration has no notion of a policy on its own.

Similar idea to Q? What's advantage?

the error analysis is also quite interesting.

And moving to Q version, and then moving to Q-learning which removes the need to knowing system dynamics, SARSA (more related to PI)

3.2 Policy Iteration

The algorithm has 2 steps:

$$\text{Find } J_{\mu^k} \text{ as a solution of } J \ \leftarrow T_{\mu^k} J \tag{7a}$$

$$\text{Get a new policy as } \mu^{k+1}(s) \leftarrow \arg\min_{u \in \mathcal{A}} \mathop{\mathbb{E}}_{\omega} \left[g(s,u,\omega) + \alpha J_{\mu^k}(f(s,u,\omega)) \right] \tag{7b}$$

Repeating these 2 steps until $J_{\mu^k}(s) = J_{\mu^{k+1}}(s)$, $\forall s$.

Theorem 3.1.

$$J_{\mu^{k+1}}(s) \leq J_{\mu^k}(s) \quad \forall s, k$$

In retrospective, J's in VI is an approximate of J^*, while J's in PI are J's of the generated policies.

In (8b), if we only run Bellman equation a finite number of times, it is call optimistic PI. If we only run that 1 time, the PI algorithm is essentially identical to VI.

$$\text{Get a new policy as } \mu^{k+1}(s) \leftarrow \arg\min_{u \in \mathcal{A}} \mathop{\mathbb{E}}_{\omega} [g(s,u,\omega) + \alpha J_k(f(s,u,\omega))] \tag{8a}$$

$$\text{Set } J_{k+1} = \widehat{J}_{k,m_k} \text{ as } \widehat{J}_{k,m+1} \leftarrow T_{\mu^k} \widehat{J}_{k,m}, \quad m = 1, \ldots, m_k - 1 \tag{8b}$$

$$\tag{8c}$$

Note that J_k is not a cost function of any μ^{k+1}.

And similarly, a version for Q?

And then, lots of alternative is variant of PI, including actor-critic, TD(λ), model-free PI, Q-learning (VI variant?)

4 todo

- Incorporate randomness to the system dynamics.
- Define proper $J^\star$
- There are 2 worries about existence: of $J^\star$ and μ^{star}.
- There are many obstacles: large state space, large action space, unknown dynamics.

The counterpart is to learn Q.

When did subscript of J go? I'd guess because of stationary assumption. So we reduce 1 order freedom.

Hey, what is policy iteration? An alternative to DP?

What if we apply DP Algorithm 1 to infinite case?

It is a nice presentation. Finite horizon, DP, then take it to the limit, infinite horizon, DP in infinite horizon becomes Bellman equation...